IMAGE PROCESSING BASED REAL TIME HARDNESS MEASUREMENT OF AN OBJECT

A PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF TECHNOLOGY

IN

ELECTRICAL ENGINEERING

By

Snigdharup Banerjee(107EE036)

&

Sanaul Hoda(107EE048)

Under the guidance of

Prof. (Dr.) Dipti Patra

Department of Electrical Engineering
National Institute of Technology
Rourkela-769008.

Department of Electrical Engineering
NATIONAL INSTITUTE OF TECHNOLOGY
Rourkela-769008.

Prof. (Dr.) Dipti Patra

B.Sc. Engg.,M.E.(Electronics Circuit &Communication System)

Ph.D (NIT, Rourkela)

CERTIFICATE

This is to certify that the project work entitled ***"Image Processing based real time hardness measurement of an object"*** *submitted by* ***Snigdharup Banerjee(107EE036)*** *&* ***Sanaul Hoda(107EE048)****, in partial fulfilment of the requirements of the award of Bachelor of Technology in the Department of Electrical Engineering at National Institute of Technology, Rourkela is an authentic work carried out by them under my supervision and guidance.*

To the best of my knowledge the matter embodied in this project work has not been submitted to any other University/Institute for award of any Degree/Diploma.

(Prof. D. Patra)

ACKNOWLEDGEMENT

*On the submission of our project report of "**Image Processing based real time hardness measurement of an object**", we would like to extend our gratitude & sincere thanks to our supervisor **Prof.(Dr.)Dipti Patra**,Department of Electrical Engineering for her constant motivation and support during the course of our work in the last one year. We truly appreciate and value her esteemed guidance and encouragement from the beginning to the end of this project work.We are indebted to her for having helped us shape the problem and providing insights towards the solution.*

We will be failing in our duty if we do not mention the laboratory staff and administrative staff of this department for their timely help. We would like to thank all whose direct and indirect support helped us completing our thesis in time.

Snigdharup Banerjee (107EE036)

Sanaul Hoda (107EE048)

PREFACE

Our project work is broadly divided into five chapters. Chapter – 1 introduces the topic and background of the existing field. Chapter – 2 deals with detailed literature review. Attempts have been made to systematically classify the available information under different sections. This chapter incorporates background information to assist in understanding the aims and results of this investigation, and also reviews recent reports by other investigators. The end of Chapter – 2 deals with the objectives of the present investigation. Chapter – 3 deals with the detailed experimental processes related to this research work. Chapter – 4 deals with the results and discussion systematically with respect to hardness measurement of soap bar,dough, a cluster of chewing gums. Chapter – 5 illustrates the possibility of Future Works. A complete list of references has been given towards the end of the thesis.

CONTENTS

LIST OF FIGURES AND TABLES

ABSTRACT

To know how useful a material can be, one needs to have a thorough understanding of its properties. Hardness is one such property whose knowledge can be put into use not only in designing machine parts, but also in many day to day applications. In this project, we have studied various hardness testing methods and have modelled a method on one of these, with slight modifications so as to enable us to compare the hardness of the objects put into daily use. We have modelled an image processing based hardness measurement technique that takes co-ordinates of the indentation as the input and gives a corresponding hardness number. For this we have used the method which involves finding the length of the diagonal of the indentation, and from this we calculate the area of the indentation. The indenting force is measured by a weighing machine, at the time of application of force. The hardness number is obtained by dividing the indenting force by the indentation area. In our project work we have compared the hardness of a soap bar, dough and a cluster of chewing gum.

CHAPTER - 1

1.0 INTRODUCTION

Ever since the dawn of civilization, humankind's insatiable appetite has led him to many path-breaking inventions and discoveries. Today's era boasts a storehouse of created means of improving the human condition that is at once thrilling and overwhelming. Yet like the fabled red shoes or the sorcerer's broom, discovery and invention will never stop. The arsenal still continues to grow. Today, thousands of new materials have come into use to revolutionize our lives. Development of new materials and their processing is helping the growth of novel technology in every field from Biology to Electronics to come into existence. Thus, knowing the properties of a material enables us to use it for our benefit. In this project we have tried to study and understand an existing method and have proposed an extrapolation of the method to help us to find a very important property of any material, that is, hardness.

With regard to materials, hardness has always been (and still is) a topic of much discussion among the engineers and the scientists. As a result we have different meanings associated with the word "hardness" . The term hardness is usually understood as the resistance offered by the material to the penetration of a harder material. Thus, hardness is the response of the material to a test load and the hardness value (or number) is generated on the basis of this response.
Depending on the method employed other information can then be determined which are due to and characterized by

- Shape and material of the indenter used
- Type and size of the load.

The hardness of materials is widely used as an important property for estimating the wear resistance and strength of machine parts and their response to heat treatment in the material and machine part manufacturing industries.

The most popular tests performed to measure indentation hardness are [5]:

- Brinell Hardness Test
- Rockwell Hardness Test
- Meyers Hardness Test
- Vickers Hardness Test.

BRINELL HARDNESS TEST

The Brinell scale characterizes the indentation hardness of materials through the scale of penetration of an indenter, loaded on a material test-piece. It is one of several definitions of hardness in materials science.

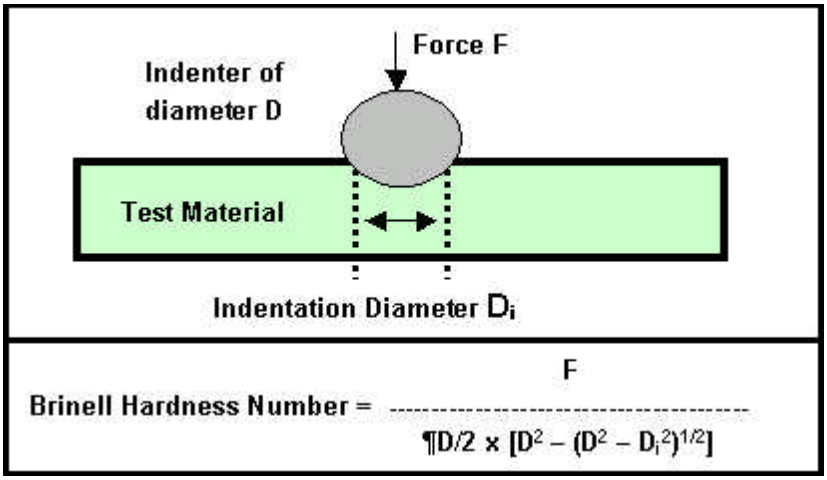


Figure 1. Brinell Hardness Testing

ROCKWELL HARDNESS TEST

The Rockwell scale is a hardness scale based on the indentation hardness of a material. The Rockwell test determines the hardness by measuring the depth of penetration of an indenter under a large load compared to the penetration made by a preload. There are different scales, which are denoted by a single letter, that use different loads or indenters. The result, which is a dimensionless number, is noted by HRX where X is the scale letter.
When testing metals, indentation hardness correlates linearly with tensile strength. This important relation permits economically important nondestructive testing of bulk metal deliveries with lightweight, even portable equipment, such as hand-held Rockwell hardness testers

The determination of the Rockwell hardness of a material involves the application of a minor load followed by a major load, and then noting the depth of penetration, vis a vis, hardness value directly from a dial, in which a harder material gives a higher number. The chief advantage of Rockwell hardness is its ability to display hardness values directly, thus obviating tedious calculations involved in other hardness measurement techniques.

MEYERS HARDNESS TEST

The Meyer hardness test is a rarely used hardness test based upon projected area of an impression. This is a more fundamental measurement of hardness than other hardness tests which are based on the surface area of an indentation. The principle behind the test is that the mean pressure required to test the material is the measurement of the hardness of the material.

The mean pressure is calculated by dividing the load by the projected area of the indentation. The result is called the Meyer hardness, which has units of megapascals(MPa).
An advantage of the Meyer test is that it is less sensitive to the applied load, especially compared to the Brinell hardness test.

VICKERS HARDNESS TEST

The Vickers hardness number[1] of a test material is defined by the surface area of the indentation made in the surface of a test specimen by a diamond pyramid indenter. Diagonal lines that indicate the indentation size are usually several micrometers to several hundred micrometers long, depending on the hardness of the material.

Image-processing techniques have a valuable practical application in the area of hardness testing of materials.

The benefits of using the image-processing principle for hardness testing are as follows:

- It helps us to eliminate the human intervention in measuring the indentation diameter, which is more prone to errors.
- It also helps us to automate the process, whereby the input of the CCD camera can be fed to desktop software to compute the results.

Present systems are extremely susceptible to the surface properties of specimens, and tests are difficult to apply to specimens other than specular-polished specimens. Moreover, we have extrapolated the concept involved in Vickers method to non-metallic, relatively softer objects.We have written a Matlab code which will directly give us the approximate hardness of the material used.Additionally, we have provided the c++ codelets for the determination of thresholding, centroid of an image, edge detection and least square method,which when used appropriately will help us achieve higher degree of accuracy.

CHAPTER - 2

2.0 LITERATURE REVIEW

The method that we have used in our project work was first proposed by **R.L.Smith** and **G.E. Sandland** in the paper titled "**An accurate method of determining the hardness testing of the materials with particular reference to those of high degree of hardness"** in 1922. In the paper titled **"Development of an automatic Vickers hardness testing system using image processing"**, by **Takao Sugimoto** and **Tadao Kawaguchi** in the year 1996, used the concept of image processing in hardness testing.

The literature review is divided into three major sections. Section 2.1 deals with the basics of image processing. Section 2.2 throws light on different components of computer vision that enables us to automate a process. The last section concerns with the principles of hardness testing method and its implementation using image processing. This chapter incorporates background information to assist in understanding the aims and results of this investigation, and also reviews recent reports by other investigators with which these results can be compared.

2.1 FUNDAMENTALS OF IMAGE PROCESSING [4]

2.1.1 IMAGE FORMATION MODEL

When an image is generated from a physical process, its values are proportional to energy radiated by a physical source.As a consequence, f(x, y) must be nonzero and finite;
i.e., $0<f(x, y)< \infty$.

The function f(x,y) may be characterized by two components:
(1) the amount of source illumination incident on the scene being viewed, and
(2) the amount of illumination reflected by the objects in the scene.

Appropriately, these are called the illumination and reflectance components and are denoted by i(x, y) and r(x, y), respectively.The two functions combine as a product to form f(x, y):
$f(x, y) = i(x, y)r(x, y)$
where

$0<i(x, y)< \infty$ and
$0<r(x, y)<1$.

The nature of i(x, y) is determined by the illumination source, and r(x, y) is determined by the characteristics of the imaged objects.

2.1.2 IMAGE SAMPLING AND QUANTIZATION

We know that there are numerous ways to acquire images, but our objective in all is the same: to generate digital images from sensed data. The output of most sensors is a continuous voltage waveform whose amplitude and spatial behavior are related to the physical phenomenon being sensed. To create a digital image, we need to convert the continuous sensed data into digital form.This involves two processes: sampling and quantization.

An image may be continuous with respect to the x- and y-coordinates, and also in amplitude.To convert it to digital form, we have to sample the function in both coordinates and in amplitude. Digitizing the coordinate values is called sampling. Digitizing the amplitude values is called quantization.

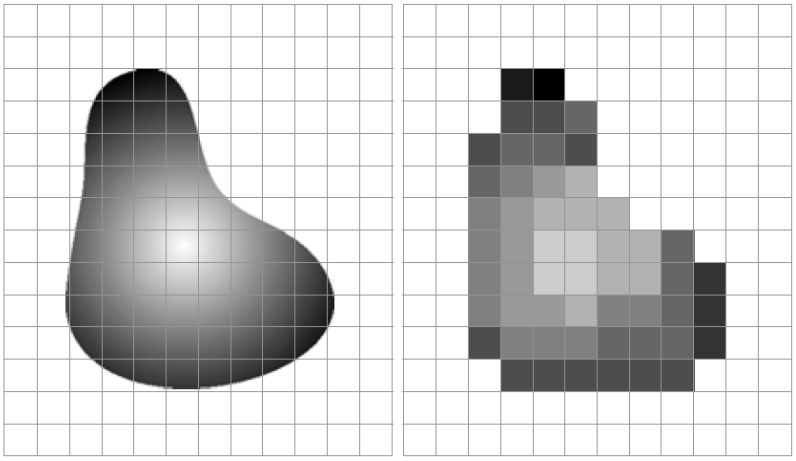


Figure 2
a)original continuous image on the sensor
b)resultant image after sampling and quantization

The result of sampling and quantization is a matrix of real numbers.Assume that an image f(x, y) is sampled so that the resulting digital image has M rows and N columns.
The values of the coordinates (x, y) now become *discrete* quantities.

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix}.$$

The right side of this equation is by definition a digital image. Each element of this matrix array is called an image element, picture element or pixel .

Sampling is the principal factor determining the *spatial resolution* of an image.
Spatial resolution is the smallest discernible detail in an image.

Gray-level resolution similarly refers to the smallest discernible change in gray level

When an actual measure of physical resolution relating pixels and the level of detail they resolve in the original scene are not necessary, it is not uncommon to refer to an L-level digital image of size M*N as having a spatial resolution of M*N pixels and a gray-level resolution of L levels.

2.1.3 NEIGHBOURS OF A PIXEL

A pixel p at coordinates (x, y) has four horizontal and vertical neighbours whose coordinates are given by (x+1, y), (x-1, y), (x, y+1), (x, y-1).
This set of pixels, called the 4-neighbours of p, is denoted by $N_4(p)$. Each pixel is a unit distance from (x, y), and some of the neighbours of p lie outside the digital image if (x, y) is on the border of the image.

The four diagonal neighbors of p have coordinates (x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1) and are denoted by $N_D(p)$. These points, together with the 4-neighbors, are called the 8-*neighbours* of p, denoted by $N_8(p)$.

2.1.4 PIXEL ADJACENCY

We consider three types of adjacency:
(a) 4-*adjacency*. Two pixels p and q with values from *V* are 4-adjacent if q is in the set $N_4(p)$.
(b) 8-*adjacency*. Two pixels p and q with values from *V* are 8-adjacent if q is in the set $N_8(p)$.
(c) *m-adjacency* (mixed adjacency).Two pixels p and q with values from *V* are m-adjacent if
(i) q is in $N_4(p)$, *or*
(ii) q is in $N_D(p)$ *and* the set $N_4(p) \cap N_4(q)$ has no pixels whose values are from *V*.
Where *V* is set of gray-level values used to define adjacency.

2.2 COMPUTER VISION

Computer vision is the science and technology of machines that see, where *see* in this case means that the machine is able to extract information from an image that is necessary to solve some task. As a scientific discipline, computer vision is concerned with the theory behind artificial systems that extract information from images. The image data can take many forms, such as video sequences, views from multiple cameras, or multi-dimensional data from a medical scanner.

As a technological discipline, computer vision seeks to apply its theories and models to the construction of computer vision systems. Examples of applications of computer vision include systems for:

- Controlling processes (e.g., an industrial robot or an autonomous vehicle).
- Detecting events (e.g., for visual surveillance or people counting).
- Organizing information (e.g., for indexing databases of images and image sequences).
- Modeling objects or environments (e.g., industrial inspection, medical image analysis or topographical modeling).
- Interaction (e.g., as the input to a device for computer-human interaction).

Computer vision is closely related to the study of biological vision. The field of biological vision studies and models the physiological processes behind visual perception in humans and other animals. Computer vision, on the other hand, studies and describes the processes implemented in software and hardware behind artificial vision systems. Interdisciplinary exchange between biological and computer vision has proven fruitful for both fields.

Computer vision is, in some ways, the inverse of computer graphics. While computer graphics produces image data from 3D models, computer vision often produces 3D models from image data. There is also a trend towards a combination of the two disciplines, e.g., as explored in augmented reality.

The organization of a computer vision system is highly application dependent. Some systems are stand-alone applications which solve a specific measurement or detection problem, while others constitute a sub-system of a larger design which, for example, also contains sub systems for control of mechanical actuators, planning, information databases, man-machine interfaces, etc. The specific implementation of a computer vision system also depends on if its functionality is pre-specified or if some part of it can be learned or modified during operation. There are, however, typical functions which are found in many computer vision systems.

- **Image acquisition**: A digital image is produced by one or several image sensors, which, besides various types of light-sensitive cameras, include range sensors, tomography devices, radar, ultra-sonic cameras, etc. Depending on the type of sensor, the resulting image data is an ordinary 2D image, a 3D volume, or an image sequence. The pixel values typically correspond to light intensity in one or several spectral bands (gray images or colour images), but can also be related to various physical measures, such as depth, absorption or reflectance of sonic or electromagnetic waves, or nuclear magnetic resonance.
- **Pre-processing**: Before a computer vision method can be applied to image data in order to extract some specific piece of information, it is usually necessary to process the data in order to assure that it satisfies certain assumptions implied by the method. Examples are
    - Re-sampling in order to assure that the image coordinate system is correct.
    - Noise reduction in order to assure that sensor noise does not introduce false information.
    - Contrast enhancement to assure that relevant information can be detected.
    - Scale-space representation to enhance image structures at locally appropriate scales.
- **Feature extraction**: Image features at various levels of complexity are extracted from the image data. Typical examples of such features are
    - Lines, edges and ridges.
    - Localized interest points such as corners, blobs or points.

  More complex features may be related to texture, shape or motion.

- **Detection/segmentation**: At some point in the processing a decision is made about which image points or regions of the image are relevant for further processing. Examples are
    - Selection of a specific set of interest points

- Segmentation of one or multiple image regions which contain a specific object of interest.

- **High-level processing**: At this step the input is typically a small set of data, for example a set of points or an image region which is assumed to contain a specific object. The remaining processing deals with, for example:

  - Verification that the data satisfy model-based and application specific assumptions.
  - Estimation of application specific parameters, such as object pose or object size.
  - Image recognition: classifying a detected object into different categories.
  - Image registration: comparing and combining two different views of the same object.

2.3 IMAGE PROCESSING BASED VICKERS HARDNESS MEASUREMENT [2]

The Vickers hardness number of a test material is defined by the surface area of the indentation made in the surface of a test specimen by a diamond pyramid indenter. Diagonal lines that indicate the indentation size are usually several micrometers to several hundred micrometers long, depending on the hardness of the material.

2.3.1 PRINCIPLES OF VICKERS HARDNESS TESTING

Vickers hardness test specimens are prepared from pieces of machine parts or materials and are usually polished to a specular finish. However, not all specimens are specular-polished, some specimens have polishing scratches, are rough-polished, or are etched for crystallographic grain size examination. Typical specimens are shown in Fig. 4. Fig. 4(a) shows an indentation on a specular-polished specimen, Fig. 4(b) shows one on a specular-polished and etched specimen, and Fig. 4(c) shows another on a rough-polished specimen. The method whereby the Vickers hardness number of specimens with these surface properties can be stably determined was studied as described below.

Vickers Hardness Number

A square-based pyramidal diamond indenter with face angles of 136° is forced with a constant load F (kg f) into the surface of a specimen as illustrated in Fig. 3. The Vickers hardness number H_V of the test material is calculated from the average projected diagonal length d (mm) of the resultant indentation by the following equation of Smith and Sandland

$$H_V = F / A \qquad \ldots(1)$$

where $A = b^2 / \sin\Theta$

$b^2 = d^2/2.$

Equation (1) is expressed by

$$H_v = 2F \sin(136^\circ/2)/d^2$$

$$= 1.5884 * F/d^2 \qquad \ldots(2)$$

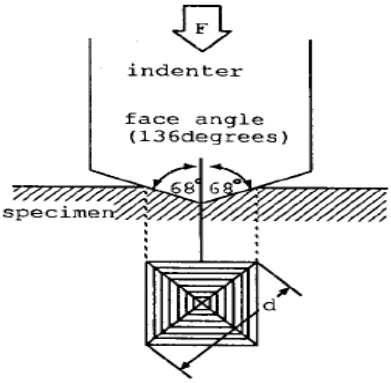


Figure 3. Principle of Vickers Hardness Testing

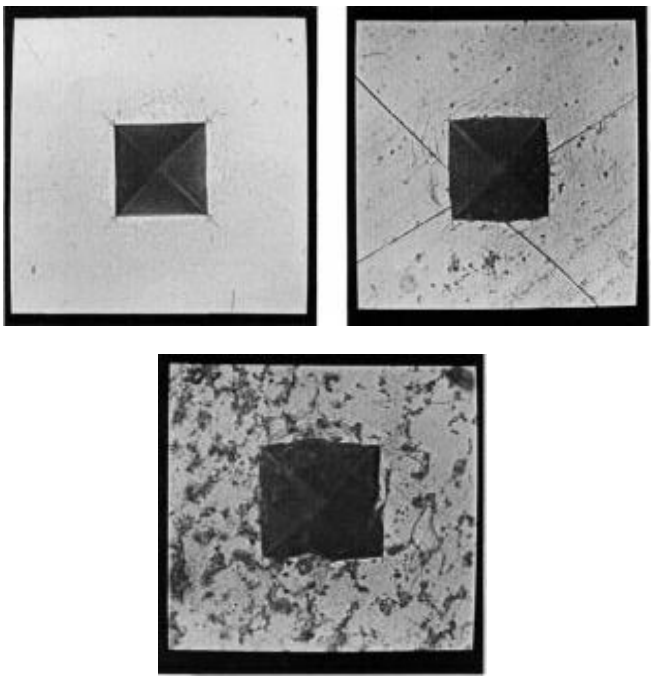


Figure 4. Indentations made on three types of specimens. (a) Indentation made on specular-polished specimen. (b) Indentation made on specular-polished and etched specimen. (c) Indentation made on rough-polished specimen.

2.3.2 CONVENTIONAL VICKERS HARDNESS TESTING METHOD, ITS PROBLEMS AND SOLUTIONS

The Vickers hardness number of the test material is the ratio of the load in kilograms-force applied to the indenter to the surface area in millimeters squared of the indentation. The indentation is so small that its average diagonal length must be determined by microscopy. Conventional automatic Vickers hardness testing systems have in the past been sensitive to the surface properties of the specimen and limited with respect to the types of specimen to which they could be applied.

Among the methods proposed for automatically measuring the indentation size are the methods involving :

- the determination of the projected area of the indentation
- mechanically moving a one-dimensional (1-D) charge-coupled device (CCD) sensor and determining the diagonal length of the indentation from the edge line slope
- assuming the edge line of the indentation to be a nearly linear curve .

Each of the above methods exhibits low measuring accuracy for rough polished specimens and low indentation measuring speed. To solve these problems,a new method is in which the visual measurement of the diagonal length of the indentation can be simulated on an image processing unit. According to this method, when the average diagonal length of the indentation is measured from the image obtained of the indentation, the threshold level of the obtained image data must be set automatically to obtain binary image data. The measured indentation size greatly depends on the setting of the threshold level. The threshold level must be determined by minimizing the effects of such factors as the specimen surface properties, indentation size and surface brightness.

CHAPTER - 3

3.0 EXPERIMENTAL PROCEDURE

Vickers hardness number is given by

$$H_v = F/A$$

where, F = force exerted by the indenter, and
A = area of the indentation.

3.1 CALCULATION OF INDENTATION AREA

To calculate the area of indentation, first the image of the indentation is taken, from which we need to calculate the length of the diagonal, and for this we have several methods available. The methods we found suitable for image processing based technique are discussed below.

3.1.1 *Calculation of Indentation Vertex Coordinates by the Least-Square Method*[2]-[3]*:*

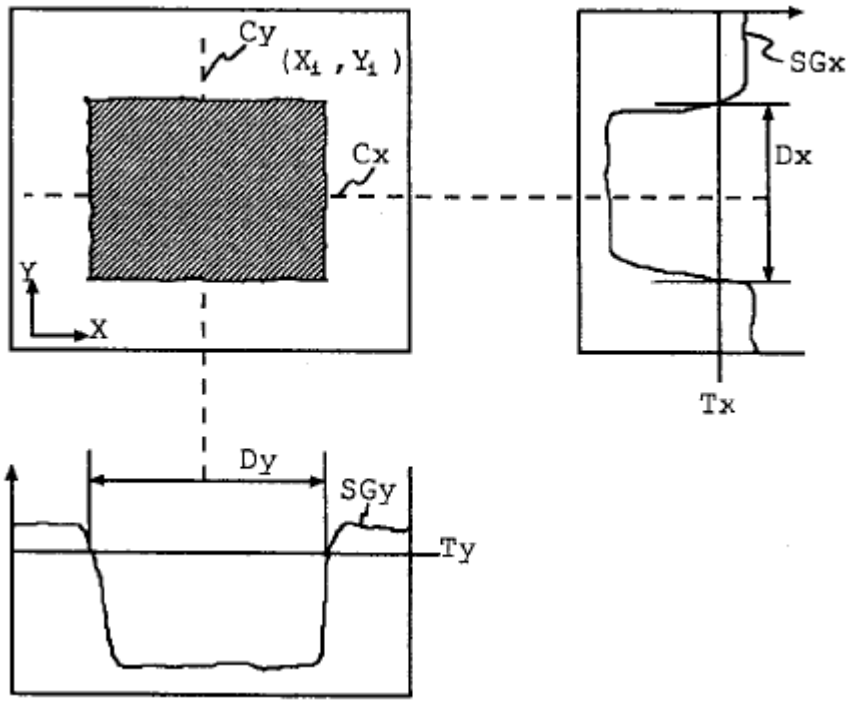


Figure 5. Procedure for determining position of indentation from image data.

The method for calculating the coordinates of the vertex of the indentation is illustrated in Fig. 5. In the binary image shown in Fig. 5, the indentation appears dark, while the specimen surface other than the indentation appears bright. The binary image data are projected in the vertical direction and designated SG_X . The binary image data are also projected in the horizontal direction and designated SG_Y . The SG_X and SG_Y data are set at suitable threshold levels T_X and T_Y , respectively. Rectangular waves corresponding to SG_X and SG_Y are obtained as a result. The approximate position of the indentation can be obtained from the rectangular waves prepared. The rectangular center lines $\mathbf{C_X}$ and $\mathbf{C_Y}$ lie on the extensions of the center lines of the rectangular waves. The center lines are extended at right angles to the vertical direction and the horizontal direction, respectively. The edges are located by the following equations from the coordinates of the points at the boundary between the dark and bright areas. If $(x_i,y_i)_{i=1,n}$ are the coordinates of each edge point, the equation of the edge line in the horizontal direction is given by

$$y = \frac{S_{xy}}{S_{xx}}x + \bar{y} - \frac{S_{xy}}{S_{xx}}\bar{x} \tag{3}$$

according to the method of linear least squares. Here,

$$S_{xy} = \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}) \tag{4}$$

$$S_{xx} = \sum_{i=1}^{n}(x_i - \bar{x})^2 \tag{5}$$

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}, \qquad \bar{y} = \frac{\sum_{i=1}^{n} y_i}{n}.$$

Similarly, the equation of the edge line in the vertical direction is given by

$$x = \frac{R_{yx}}{R_{yy}}y + \bar{x} - \frac{R_{yx}}{R_{yy}}\bar{y} \tag{6}$$

Where

$$R_{yx} = \sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x}) \tag{7}$$

$$R_{yy} = \sum_{i=1}^{n}(y_i - \bar{y})^2. \tag{8}$$

If the two primarily approximated lines given by (3) and (6)

$$y = a_1x + b_1 \tag{9}$$

$$x = a_2y + b_2 \tag{10}$$

the coordinates of their intersection are given by

$$(x, y) = \left(\frac{a_2 b_1 + b_2}{1 - a_1 a_2}, \frac{a_1 b_2 + b_1}{1 - a_1 a_2} \right) \qquad (11)$$

$$a_1 = (S_{xy}/S_{xx}), \qquad b_1 = \bar{y} - (S_{xy}/S_{xx})$$
$$a_2 = (R_{yx}/R_{yy}), \qquad b_2 = \bar{x} - (R_{yx}/R_{yy}).$$

Equation (11) can be used to determine the vertex coordinates of the indentation.

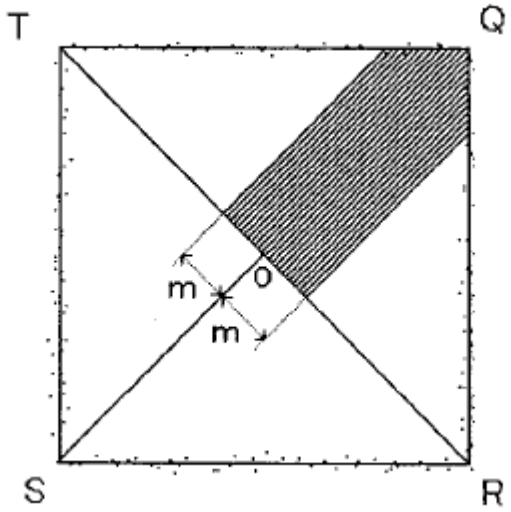


Figure 6. Corner scanning method.

3.1.2 *Calculation of Indentation Vertex Coordinates by Corner Scanning Method* [2]*:*

In Fig. 6, Q, R, S, and T are the indentation vertices calculated by the method (1) above. The diagonals QOS and TOR are constructed from the vertices. The intersection of the two diagonals is O. The distance between the two nearest pixels $b_1, b_2, \ldots, b_m$, spaced apart by an appropriate distance m on the diagonal lines from the intersection O, are drawn perpendicular to the diagonal lines from each diagonal line to each edge line. The end of the longest segment is taken as the vertex of the indentation. This method can be applied to each vertex to determine the vertex coordinates of the indentation.

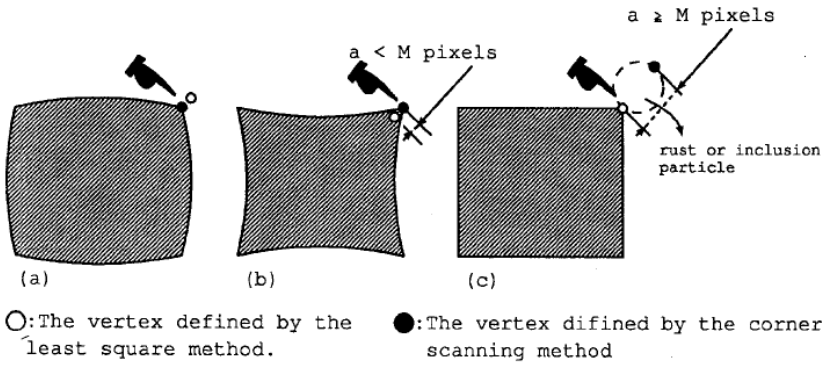


Figure 7. Comparison of vertex defined by the two methods

***Selection of Vertex Coordinates**:*

The indentation may be approximately square shaped, barrel shaped as shown in Fig. 7(a), bobbin shaped as shown in Fig. 7(b), or may have rust or inclusion particles near a vertex as shown in Fig. 7(c). In the latter three cases, the vertices of the indentation determined by (1) and (2) above may not be clearly defined. To improve the agreement of the calculated vertex coordinates with those measured by the conventional visual testing method, selection of the indentation coordinates calculated by the above methods (1) and (2) was studied in the development stage for the three shapes. The following two cases illustrate selection criteria.

Case a: When the vertex coordinates calculated by the least square method are outside of those obtained by the corner scanning method. When the test specimen has a low hardness, its plastic deformation is so large that the indentation may be shaped as shown in Fig. 7(a). In this case, the indentation vertex coordinates determined by the least square method are outside of those obtained by the corner scanning method, and the vertex coordinates obtained by the corner scanning method are adopted.

Case b: When the vertex coordinates calculated by the least-square method are inside of those obtained by the corner scanning method .When the test specimen has a high hardness, its elastic deformation is so large that the indentation may be shaped like a bobbin as shown in Fig. 7(b), and the vertex coordinates calculated by the least-square method are inside those obtained by the corner scanning method. If there is rust or inclusion particles near the indentation as shown in Fig. 7(c), the vertex coordinates calculated by the least square method are also inside of those obtained by the corner scanning method. When a<M, the vertex coordinates obtained by the corner scanning method are adopted. When a>M, the vertex coordinates obtained by the least square method are adopted. The value “M” is the distance between the coordinates obtained by the least square method and the coordinates obtained by the corner scanning method. The value “M” is an experimentally determined constant of such a magnitude that corresponds to a few pixels to tens of pixels.

3.1.3 *Calculation of area by counting the number of pixels using image segmentation technique .*

First the image of the indentation is taken. The indentation appears darker than the rest of the material surface. Then the image is binarized by setting the darker pixels to 0 (black) and the

brighter pixels to 255 (white).After that the image is segmented and the number of dark pixels are counted. This will directly give the area of the indentation.

3.2 EXPERIMENTAL APPARATUS

During this research work, several equipments were used. The list of equipments used along with their purpose and specification are as follows.

Table 1. List of Equipments

SL. NO.	EQUIPMENT	SPECIFICATION	PURPOSE
1	Indenter	Avg Tip angle = 73˚	To make indentation on the test object.
2	Weighing machine	Max= 6 kg	To measure the force exerted on the test object by the indenter.
3	Camera	12.1MegaPixels Optical zoom=3X DSC-S2100	To take a high resolution image of the indentation.
4	Test materials	Materials with varied hardness values (e.g. soap bar, dough, a cluster of chewing gum)	To study the relative hardness of these test materials.

Figure 8. The indenter that we have used.

3.3 HARDWARE LIMITATIONS AND ASSUMPTIONS

Due to the lack of availability and access to proper equipments, we had to make do with their inferior substitutes, though not accurate, helped us to make a satisfactory study of the relative hardness of the test materials. The hardware limitations and requisite assumptions are:

- In actual Vickers hardness testing method a diamond pyramid indenter is used, but as we were dealing with relatively softer objects so we have used a wooden indenter whose tip angle is accordingly modified.
- Due to unavailability of proper force measuring apparatus, we used a weighing machine with a least count of 0.1g only and maximum limit of 6 kg.
- Had to take images with a 12.1MP camera as we did not have one of higher resolution.
- As the calibration of pixel size was difficult to achieve with the available resources we had to express our area in $(pixel)^2$.
- We did not have a provision for obtaining the image of indentation in real time, so we used test materials that would retain their indentation even after the removal of applied force.
- We have used materials that have hardness comparable to that of human body tissues, as we aspire to develop a technique that will measure the hardness of such delicate objects.

3.4 SOME IMPORTANT STEPS IN IMAGE PROCESSING AND THEIR MATLAB/C++ CODES [6]

3.4.1 THRESHOLDING

Thresholding is an image processing technique for converting a grayscale or color image to a binary image based upon a threshold value. If a pixel in the image has an intensity value less than the threshold value, the corresponding pixel in the resultant image is set to black. Otherwise, if the pixel intensity value is greater than or equal to the threshold intensity, the resulting pixel is set to white. Thus, creating a binary image, or an image with only 2 colors, black (0) and white (255). Image thresholding is very useful for keeping the significant part of an image and getting rid of the unimportant part or noise. This holds true under the assumption that a reasonable threshold value is chosen.

How to choose a threshold value?. In our case, our histogram is going to consist of the vertical axis being a ratio between a pixel intensity value of "x" to the total number of pixels in an image. The horizontal axis will contain the pixel value "x". A grayscale image and its corresponding histogram are seen below to clear up any misconceptions.

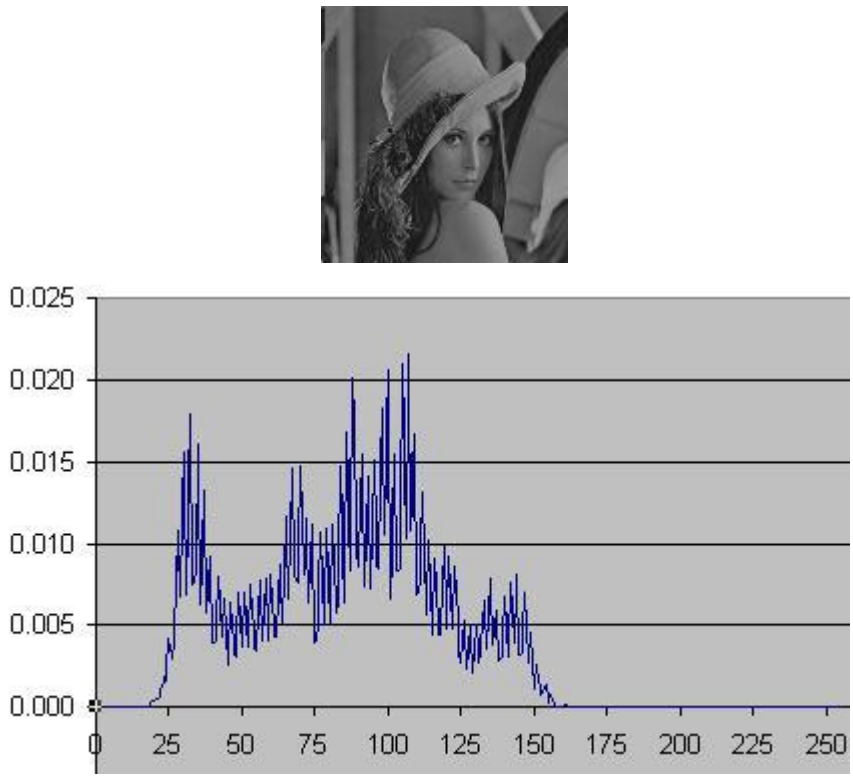


Figure 9. Histogram of the test image 1

Looking at the picture above, one can see that there are no white pixel intensity values (255). And we can see that the histogram coincides with this observation. So rather than using trial-and-error to select an accurate threshold value, one can see from the histogram above that the most dominant pixel value, or the one occurring most throughout the image, is 107. So if we select a threshold value of 107 and make every pixel with an intensity below the threshold black, and every pixel with an intensity above the threshold white, we come up with the following binary image:

Figure 10. Binarized test image 1

```
clear all
[tmp,idx]=imread('lena.bmp');
a=ind2gray(tmp,idx); % gray scale image of lena, value between 0 and 1
clear tmp idx
figure(1),clf,colormap('gray'),imshow(a),title('original Lena image')
[m,n]=size(a); % size of image a
```

```
b=reshape(a,m*n,1); % into a column vector
figure(2),hist(b,50),title('histogram of image')
% first do global thresholding
mu=rand(2,1); % value betwen 0 and 1, two clusters only
[W,iter,Sw,Sb,Cova]=kmeansf(b,mu);% W is the mean,
% Cova is the covariance matrices
% member: membership of each X:  K by 1 vector of elements 1 to c
[d,member]=kmeantest(b,sort(W));
c=reshape(member-1,m,n);
clear d member b
figure(3),clf,colormap('gray'),imshow(c)
title('global threshold')

% next do local threshold, partition the image into 64 x 64 blocks
% and do threshold within each block
c=zeros(512,512); trials=0;
for i=1:8,
   for j=1:8,
      trials=trials+1;
      disp([int2str(trials) ' of 64 iterations ...']);
      tmp=a(64*(i-1)+1:64*i,64*(j-1)+1:64*j);
      tmpi=reshape(tmp,64*64,1);
      mu=sort(rand(2,1)); % value betwen 0 and 1, two clusters only
      [W,iter,Sw,Sb,Cova]=kmeansf(tmpi,mu);% W is the mean,
      % Cova is the covariance matrices
      % member: membership of each X:  K by 1 vector of elements 1 to c
      [d,member]=kmeantest(tmpi,sort(W));
      c(64*(i-1)+1:64*i,64*(j-1)+1:64*j)=reshape(member,64,64);
   end
end
figure(4),clf,colormap('gray'),imshow(c-1),
title('local threshold, 64x64 block');
```

3.4.2 CENTROID OF AN IMAGE

To find the centroid of an image, the image first has to be binarized. The centroid program will then calculate the centroid based on where the majority of the black pixels are located. If you have a completely black 20 x 20 BMP, then the centroid would be located in the exact center of that square. However, if you have an image like the one below, the centroid would be located in the center of the black square located in the top left corner.

Figure 11. Test image 2

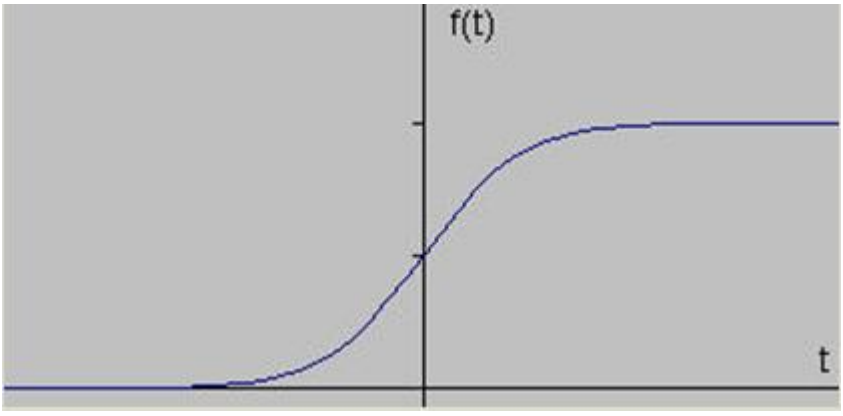
```
#include (stdio.h)
#include (stdlib.h)
#include (math.h)

typedef struct{float row; float col; float area;} imageCentroid;
typedef struct{int rows; int cols; unsigned char *data;} sImage;

long getImageInfo(FILE*, long, int);

int main(int argc, char* argv[])
{
  FILE *bmpInput;
  sImage originalImage;
  unsigned char someChar;
  unsigned char *pChar;
  imageCentroid ic;
  int r, c, nColors;

  someChar = '0';
  pChar = &someChar;

  if(argc < 2)
  {
    printf("Usage: %s bmpInput.bmp\n", argv[0]);
    exit(0);
  }
  printf("Reading filename %s\n", argv[1]);

  bmpInput = fopen(argv[1], "rb");
  fseek(bmpInput, 0L, SEEK_END);

  ic.row = ic.col = ic.area = 0.0;

  /*-------GET INPUT BMP DATA---------*/
  originalImage.cols = getImageInfo(bmpInput, 18, 4);
  originalImage.rows = getImageInfo(bmpInput, 22, 4);
  nColors = getImageInfo(bmpInput, 46, 4);

  fseek(bmpInput, (54 + 4*nColors), SEEK_SET);

  for(r=0; r<=originalImage.rows-1; r++)
  {
    for(c=0; c<=originalImage.cols-1; c++)
    {
      fread(pChar, sizeof(char), 1, bmpInput);
      if(*pChar == 0.0)
      {
        ic.row = ic.row + (originalImage.rows-1) - r;
        ic.col = ic.col + c;
        ic.area = ic.area + 1.0;
      }
    }
  }

  printf("Sum of black row pixels = %f\n", ic.row);
  printf("Sum of black col pixels = %f\n", ic.col);

  ic.row = ic.row/ic.area;
  ic.col = ic.col/ic.area;
```

```
  printf("Centroid location:\n");
  printf("row = %f\n", ic.row);
  printf("column = %f\n", ic.col);
}
```

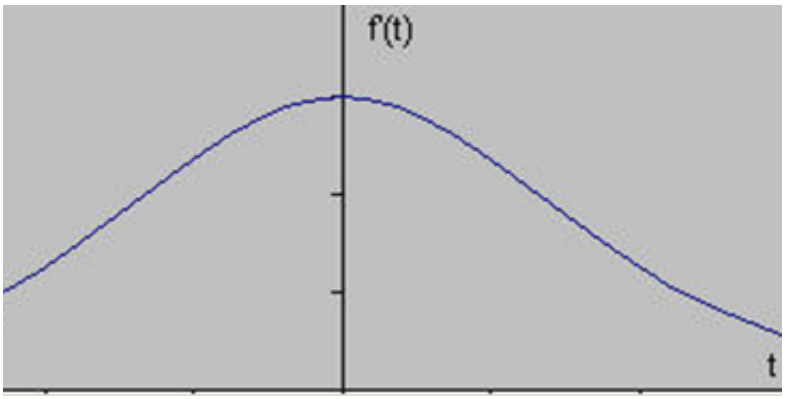
3.4.3 EDGE DETECTION USING SOBEL METHOD.

Figure 12. Test image for edge detection

Edges characterize boundaries and are therefore a problem of fundamental importance in image processing. Edges in images are areas with strong intensity contrasts – a jump in intensity from one pixel to the next. Edge detecting an image significantly reduces the amount of data and filters out useless information, while preserving the important structural properties in an image**.** There are many ways to perform edge detection. However, the majority of different methods may be grouped into two categories, gradient and Laplacian. The gradient method detects the edges by looking for the maximum and minimum in the first derivative of the image. The Laplacian method searches for zero crossings in the second derivative of the image to find edges. An edge has the one-dimensional shape of a ramp and calculating the derivative of the image can highlight its location. Suppose we have the following signal, with an edge shown by the jump in intensity below:



If we take the gradient of this signal (which, in one dimension, is just the first derivative with respect to t) we get the following:



Clearly, the derivative shows a maximum located at the center of the edge in the original signal. This method of locating an edge is characteristic of the "gradient filter" family of edge detection filters and includes the Sobel method. A pixel location is declared an edge location if the value of the gradient exceeds some threshold. As mentioned before, edges will have higher pixel intensity values than those surrounding it. So once a threshold is set, you can compare the gradient value to the threshold value and detect an edge whenever the threshold is exceeded.

SOBEL

Based on this one-dimensional analysis, the theory can be carried over to two-dimensions as long as there is an accurate approximation to calculate the derivative of a two-dimensional image. The Sobel operator performs a 2-D spatial gradient measurement on an image. Typically it is used to find the approximate absolute gradient magnitude at each point in an input grayscale image. The Sobel edge detector uses a pair of 3x3 convolution masks, one estimating the gradient in the x-direction (columns) and the other estimating the gradient in the y-direction (rows). A convolution mask is usually much smaller than the actual image. As a result, the mask is slid over the image, manipulating a square of pixels at a time. The actual Sobel masks are shown below:

-1	0	+1
-2	0	+2
-1	0	+1

Gx

+1	+2	+1
0	0	0
-1	-2	-1

Gy

The magnitude of the gradient is then calculated using the formula:

$$|G| = \sqrt{Gx^2 + Gy^2}$$

An approximate magnitude can be calculated using:

|G| = |Gx| + |Gy|

The code for the Sobel edge detector is shown below and uses the above gradient approximation.

```
#include (stdio.h)
#include (stdlib.h)
#include (math.h)
#include (alloc.h)

/*-------STRUCTURES---------*/
typedef struct {int rows; int cols; unsigned char* data;} sImage;

/*-------PROTOTYPES---------*/
long getImageInfo(FILE*, long, int);
void copyImageInfo(FILE* inputFile, FILE* outputFile);
void copyColorTable(FILE* inputFile, FILE* outputFile, int nColors);

int main(int argc, char* argv[])
{
   FILE *bmpInput, *bmpOutput;
   sImage      originalImage;
   sImage      edgeImage;
   unsigned int X, Y;
   int I, J;
   long sumX, sumY;
   int nColors, SUM;
   unsigned long vectorSize;
```

```
unsigned long fileSize;
int GX[3][3];
int GY[3][3];
unsigned char *pChar, someChar;
unsigned int row, col;

someChar = '0'; pChar = &someChar;

/* 3x3 GX Sobel mask.  Ref: www.cee.hw.ac.uk/hipr/html/sobel.html */
GX[0][0] = -1; GX[0][1] = 0; GX[0][2] = 1;
GX[1][0] = -2; GX[1][1] = 0; GX[1][2] = 2;
GX[2][0] = -1; GX[2][1] = 0; GX[2][2] = 1;

/* 3x3 GY Sobel mask.  Ref: www.cee.hw.ac.uk/hipr/html/sobel.html */
GY[0][0] =  1; GY[0][1] =  2; GY[0][2] =  1;
GY[1][0] =  0; GY[1][1] =  0; GY[1][2] =  0;
GY[2][0] = -1; GY[2][1] = -2; GY[2][2] = -1;

if(argc < 2) {
  printf("Usage: %s bmpInput.bmp\n", argv[0]);
  exit(0);
};
printf("Reading filename %s\n", argv[1]);

/*-------DECLARE INPUT & OUTPUT FILES-------*/
bmpInput = fopen(argv[1], "rb");
bmpOutput = fopen("edgeSob.bmp", "wb");

/*---SET POINTER TO BEGINNING OF FILE----*/
fseek(bmpInput, 0L, SEEK_END);

/*-------GET INPUT BMP DATA--------*/
fileSize = getImageInfo(bmpInput, 2, 4);
originalImage.cols = (int)getImageInfo(bmpInput, 18, 4);
originalImage.rows = (int)getImageInfo(bmpInput, 22, 4);
edgeImage.rows = originalImage.rows;
edgeImage.cols = originalImage.cols;

/*--------PRINT DATA TO SCREEN----------*/
printf("Width: %d\n", originalImage.cols);
printf("Height: %d\n", originalImage.rows);
printf("File size: %lu\n", fileSize);

nColors = (int)getImageInfo(bmpInput, 46, 4);
printf("nColors: %d\n", nColors);

/*------ALLOCATE MEMORY FOR FILES--------*/
vectorSize = fileSize - (14+40+4*nColors);
printf("vectorSize: %lu\n", vectorSize);
edgeImage.data = farmalloc(vectorSize*sizeof(unsigned char));
if(edgeImage.data == NULL) {
     printf("Failed to malloc edgeImage.data\n");
     exit(0);
}
printf("%lu bytes malloc'ed for edgeImage.data\n", vectorSize);

originalImage.data = farmalloc(vectorSize*sizeof(unsigned char));
if(originalImage.data == NULL) {
     printf("Failed to malloc originalImage.data\n");
     exit(0);
}
```

```
   printf("%lu bytes malloc'ed for originalImage.datt\n", vectorSize);

   /*------COPY HEADER AND COLOR TABLE---------*/
   copyImageInfo(bmpInput, bmpOutput);
   copyColorTable(bmpInput, bmpOutput, nColors);
   fseek(bmpInput, (14+40+4*nColors), SEEK_SET);
   fseek(bmpOutput, (14+40+4*nColors), SEEK_SET);

   /* Read input.bmp and store it's raster data into originalImage.data */
   for(row=0; row<=originalImage.rows-1; row++) {
        for(col=0; col<=originalImage.cols-1; col++) {
             fread(pChar, sizeof(char), 1, bmpInput);
             *(originalImage.data + row*originalImage.cols + col) = *pChar;
        }
   }

   /*-------------------------------------------------
               SOBEL ALGORITHM STARTS HERE
   -------------------------------------------------*/
   for(Y=0; Y<=(originalImage.rows-1); Y++)  {
        for(X=0; X<=(originalImage.cols-1); X++)  {
             sumX = 0;
             sumY = 0;

             /* image boundaries */
             if(Y==0 || Y==originalImage.rows-1)
                  SUM = 0;
             else if(X==0 || X==originalImage.cols-1)
                  SUM = 0;

             /* Convolution starts here */
             else   {

               /*-------X GRADIENT APPROXIMATION------*/
               for(I=-1; I<=1; I++)  {
                   for(J=-1; J<=1; J++)  {
                      sumX = sumX + (int)( (*(originalImage.data + X + I +
                              (Y + J)*originalImage.cols)) * GX[I+1][J+1]);
                   }
               }

               /*-------Y GRADIENT APPROXIMATION-------*/
               for(I=-1; I<=1; I++)  {
                   for(J=-1; J<=1; J++)  {
                       sumY = sumY + (int)( (*(originalImage.data + X + I +
                               (Y + J)*originalImage.cols)) * GY[I+1][J+1]);
                   }
               }

               /*---GRADIENT MAGNITUDE APPROXIMATION (Myler p.218)----*/
               SUM = abs(sumX) + abs(sumY);
             }

             if(SUM>255) SUM=255;
             if(SUM<0) SUM=0;

             *(edgeImage.data + X + Y*originalImage.cols) = 255 - (unsigned
char)(SUM);
             fwrite((edgeImage.data + X +
Y*originalImage.cols),sizeof(char),1,bmpOutput);
        }
```

```
   }

   printf("See edgeSob.bmp for results\n");
   fclose(bmpInput);
   fclose(bmpOutput);
   farfree(edgeImage.data);      /* Finished with edgeImage.data */
   farfree(originalImage.data);  /* Finished with originalImage.data */
   return 0;
}

/*----------GET IMAGE INFO SUBPROGRAM--------------*/
long getImageInfo(FILE* inputFile, long offset, int numberOfChars)
{
  unsigned char *ptrC;
  long value = 0L;
  unsigned char dummy;
  int i;

  dummy = '0';
  ptrC = &dummy;

  fseek(inputFile, offset, SEEK_SET);

  for(i=1; i<=numberOfChars; i++)
  {
    fread(ptrC, sizeof(char), 1, inputFile);
    /* calculate value based on adding bytes */
    value = (long)(value + (*ptrC)*(pow(256, (i-1))));
  }
  return(value);

} /* end of getImageInfo */

/*-------------COPIES HEADER AND INFO HEADER----------------*/
void copyImageInfo(FILE* inputFile, FILE* outputFile)
{
  unsigned char *ptrC;
  unsigned char dummy;
  int i;

  dummy = '0';
  ptrC = &dummy;

  fseek(inputFile, 0L, SEEK_SET);
  fseek(outputFile, 0L, SEEK_SET);

  for(i=0; i<=50; i++)
  {
    fread(ptrC, sizeof(char), 1, inputFile);
    fwrite(ptrC, sizeof(char), 1, outputFile);
  }

}

/*----------------COPIES COLOR TABLE-----------------------------*/
void copyColorTable(FILE* inputFile, FILE* outputFile, int nColors)
{
  unsigned char *ptrC;
  unsigned char dummy;
  int i;
```

```
  dummy = '0';
  ptrC = &dummy;

  fseek(inputFile, 54L, SEEK_SET);
  fseek(outputFile, 54L, SEEK_SET);

  for(i=0; i<=(4*nColors); i++)  /* there are (4*nColors) bytesin color
table */
  {
    fread(ptrC, sizeof(char), 1, inputFile);
    fwrite(ptrC, sizeof(char), 1, outputFile);
  }

}
```

SOBEL EXPLANATION

The mask is slid over an area of the input image, changes that pixel's value and then shifts one pixel to the right and continues to the right until it reaches the end of a row. It then starts at the beginning of the next row. The example below shows the mask being slid over the top left portion of the input image represented by the green outline. The formula shows how a particular pixel in the output image would be calculated. The center of the mask is placed over the pixel you are manipulating in the image. And the I & J values are used to move the file pointer so you can mulitply, for example, pixel (a22) by the corresponding mask value (m22). It is important to notice that pixels in the first and last rows, as well as the first and last columns cannot be manipulated by a 3x3 mask. This is because when placing the center of the mask over a pixel in the first row (for example), the mask will be outside the image boundaries.

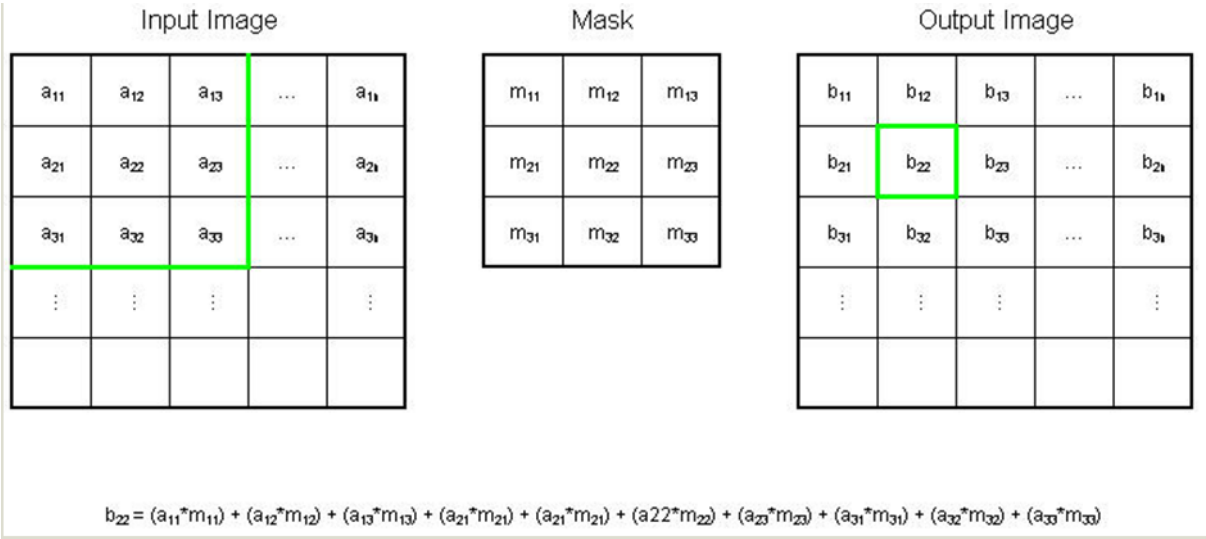


Figure 13. Mask for edge detection

The GX mask highlights the edges in the horizontal direction while the GY mask highlights the edges in the vertical direction. After taking the magnitude of both, the resulting output detects edges in both directions.

3.4.4 C++ CODE TO FIND THE VERTICES OF AN IMAGE USING LEAST SQUARE METHOD

```
#include<stdio.h>
int main()
{
 int n, i, pid, start, size, nproc, shmid1, shmid2, sh3, sh4, sh5, sh6, *y;
 int *x, *X, *XY, *X2, *Y, y;
        float a, b, data;

        printf("Enter the no.of pixels :: ");
        scanf("%d",&size);

        printf("Enter the first pixel co-ordinates :: ");
        scanf("%d",&start);
        n=start;

        y=(int*)create_memory(size*2,&shmid1);
        x=(int*)create_memory(size*2, &shmid2);
        X=(int*)create_memory(size*2,&sh6);
        XY=(int*)create_memory(2,&sh3);
        X2=(int*)create_memory(2, &sh4);
        Y=(int*)create_memory(2,&sh5);

        *Y=*XY=*X2=0;
        for(i=0;i<size;i++)
        {
                x[i]=start;
                printf("Enter the data for pixels %d :: ",x[i]);
                scanf("%d",&y[i]);
                start++;
        }


        n=(int)(x[size-1]-x[0])/2;
        n=x[n];

        for(i=0;i<size;i++)
                X[i]=x[i]-n;


        printf("Enter the total number of pixels :: ");
        scanf("%d",&nproc);

        pid=create_process(&nproc);

        for(i=pid;i<size;i+=nproc)
        {
                *XY += X[i]*y[i];
```

```
                *X2 += (X[i])*(X[i]);
                *Y += y[i];
        }

        join_process(&nproc,&pid);

        a=(float)*XY/(*X2);
        b=(float)*Y/size;

        printf("The equation is y = %.2fx + %.2f",a,b);
        data=a*(y-n) + b;
        printf("\n %d, the data value is %0.2f",y,data);

        printf("\n");

        return 0;
}
```

3.4.5 IMAGE PROCESSING BASED MATLAB CODE DEVELOPED FOR DETERMINATION OF HARDNESS NUMBER OF AN OBJECT

imread(filename,fmt);

This command reads a grayscale or color image from the file specified by the string *filename*. The text string *fmt* specifies the format of the file by its standard file extension.

imtool(filename,fmt);

This command opens a new image tool in an empty state, giving us directly the pixel co-ordinates and the corresponding intensity value.

MATLAB CODE

```
clear all
i=imread('image.JPG');
% for soap bar i=imread('soap.JPG');
% for dough i=imread('dough.JPG');
% for a cluster of chewing gum i=imread('gum.JPG');
n = input('Enter no of pixels: ');
display('Enter pixel coordinates: ')
for i = 1:1:n
    x(i) = input('');
    y(i) = input('');
end
format('short')
xbar = sum(x)/n;
ybar = sum(y)/n;
Sxy = 0;
Sxx = 0;
Ryx = 0;
Ryy = 0;
```

```
for j = 1:1:n
    Sxy = Sxy+(x(i)-xbar)*(y(i)-ybar);
    Sxx = Sxx+(x(i)-xbar)^2;
    Ryx = Ryx+(y(i)-ybar)*(x(i)-xbar);
    Ryy = Ryy+(y(i)-ybar)^2;
end
a1 = Sxy/Sxx;
b1 = ybar-a1;
a2 = Ryx/Ryy;
b2 = xbar-a2;
% coordinates of point of intersection
xc = (a2*b1+b2)/(1-a1*a2);
yc = (a1*b2+b1)/(1-a1*a2);
display('Enter one of the corner pixel coordinates')
xcc = input('');
ycc = input('');
%distance between the corner coordinate and centroid coordinates
d = sqrt((xcc-xc)^2+(ycc-yc)^2);
A = (2*d)^2/(2*sin(0.637))/(1*(10^38));
F = input('Enter the force applied: ');
H = F/A;
display('Hardness Number = ');
disp(H);
```

CHAPTER - 4

4.0 RESULTS AND DISCUSSIONS

4.1 RESULTS

On running the above Matlab code, we obtain results for different specimens as follows:

4.1.1 SOAP BAR

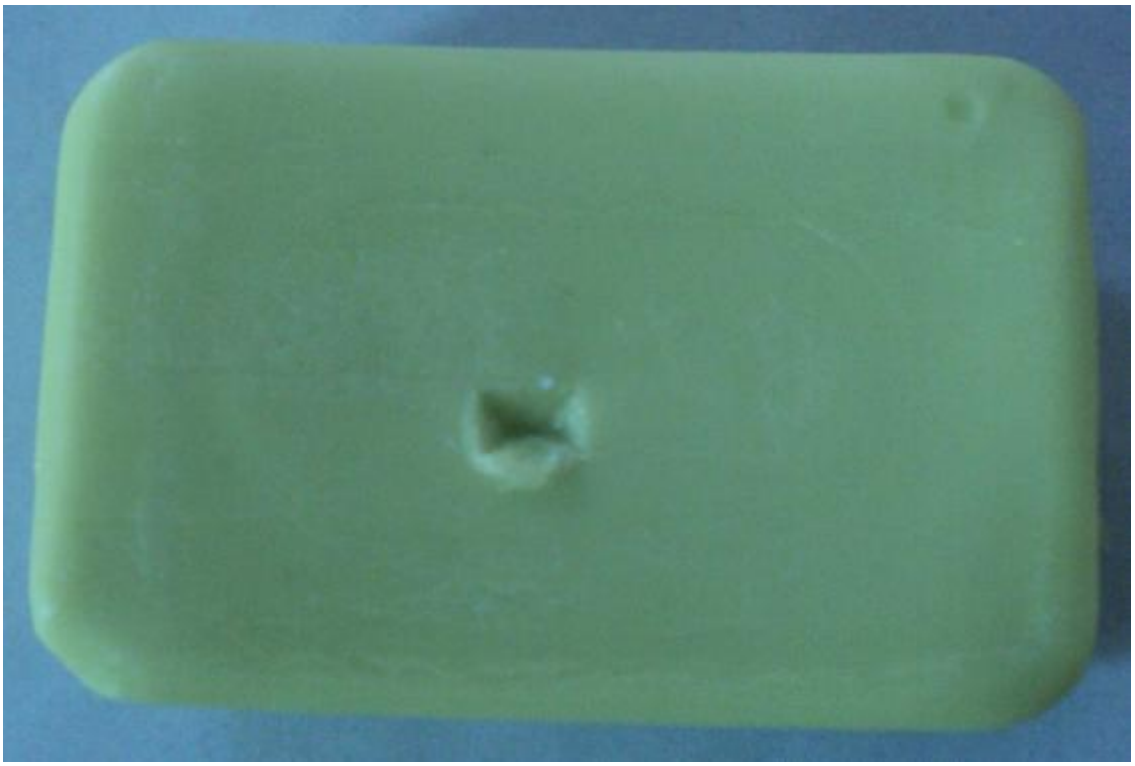


Figure 14. Test specimen-1 (soap bar)

```
Enter no of pixels: 10
Enter pixel coordinates:
1799 1951
1799 1895
1795 1847
1795 1819
```

```
1835 1815

1935 1815

1987 1819

2007 1855

2015 1903

2015 1950

Enter one of the corner pixel coordinates

1799 1951

Enter the force applied: 5076.3

Hardness Number =

  241.90
```

4.1.2 DOUGH

Figure 15. Test specimen-2 (dough)

```
Enter no of pixels: 14

Enter pixel coordinates:

1199 1127
```

```
1223 1347

1227 1547

1235 1675

1279 1891

1327 1987

1491 1987

1727 2003

2067 2039

2347 2003

2451 1475

2427 1027

1879 1027

1539 1019

Enter one of the corner pixel coordinates

1199 1127

Enter the force applied: 72.2

Hardness Number =

  8.4191
```

4.1.3 A CLUSTER OF CHEWING GUMS

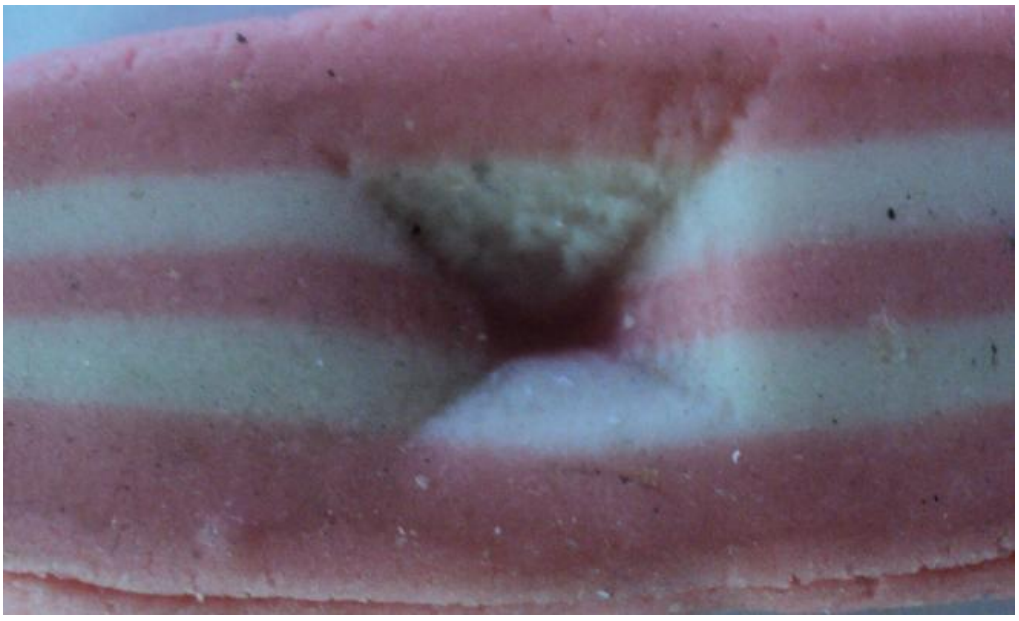


Figure 16. Test specimen-3 (chewing gum)

```
Enter no of pixels: 12
Enter pixel coordinates:
1347 1283
1567 1231
1895 1211
2123 1187
2287 1191
2543 1223
2475 1519
2455 1815
2471 2083
2095 2143
1567 2179
1383 1779
Enter one of the corner pixel coordinates
1347 1283
Enter the force applied: 757.5
Hardness Number =
   33.0276
```

4.2 DISCUSSIONS

From the above output , we have found the hardness number as such

Table 2. Final Result

SL. NO.	**SPECIMEN**	**FORCE EXERTED(g)**	**HARDNESS NUMBER**
1	SOAP BAR	5076.3	**241.90**
2	DOUGH	72.2	**8.4191**
3	CHEWING GUM	757.5	**33.0276**

The above table shows us that soap bar is harder than the chewing gum, which is harder than the dough. This result agrees with the actual relative hardness of these three objects.

CHAPTER - 5

5.0 SCOPE FOR FUTURE WORK

Cancer is a class of disease in which a group of cells exhibit uncontrolled growth, invasion that intrudes upon and destroys adjacent tissues, and sometimes metastasis, or spreading to other locations in the body through blood or lymph. The presence of cancer can be detected on the basis of symptoms, or findings on radiology. Precise diagnosis of cancer, however, requires the microscopic examination of a biopsy specimen. Most cancers can be treated. Available treatments include chemotherapy, radiotherapy and surgery. Our method will help in localizing abnormalities at the physician's fingertips. This will help to determine the hardness of a suspicious mass for the detection of tumors in organs such as the breast, thyroid, prostate and liver. Many people wonder if small bumps or lumps on the body are something to worry about. Most lumps and swellings are benign (not cancerous) and are innocuos, especially the kind that feel soft and roll easily.

However there are certain hurdles that need to be taken care of such as we require a device that can be used to make an indentation on the organ under consideration, inside the body. Moreover, an accompanying force measuring apparatus is required, capable of measuring forces of very low magnitude. If these concerns are addressed then this method will prove fruitful in early detection of cancer, and will eventually help in bringing down cancer mortality rate.

REFERENCES

[1] R.L.Smith and G.E.Sandland, "An accurate method of determining the hardness testing of the materials with particular reference to those of high degree of hardness", in *Proc. Inst. Mechanical Engineers*, 1922, pt. 1, pp. 623-641.

[2] T.Sugimoto and T.Kawaguchi, "Development of an automatic Vickers Hardness Testing System using Image Processing technology",in *IEEE Transactions on Industrial Electronics*, Vol. 44, No. 5, 1997.

[3] T.Sugimoto, Y. Fujikake, and T. Nishimura, "Indentation Hardness Tester", U.S.Patent 07 807 734.

[4] R.C.Gonzalez and R.E.Woods, *Digital Image Processing*, 2nd Ed, Prentice Hall,(ISBN-0201 180758),2002.

[5] Sagar Limaye, *Application of Imaging in Hardness Measurement*, Patni Computer Systems Ltd.

[6] http://www.pages.drexel.edu/~weg22/hist_thresh_cent.html

http://www.pages.drexel.edu/~weg22/edge.html

[7] www.emedicinehealth.com

[8] http://en.wikipedia.org/wiki/Brinell_scale

http://en.wikipedia.org/wiki/Rockwell_scale

http://en.wikipedia.org/wiki/Computer_vision

[9] http://homepages.cae.wisc.edu/~ece533/matlab/threshdemo.m